Chapter 1
Introduction to HTML & Intermediate HTML

- Hitting Tab after typing html in Sublime, creates the boilerplate automatically for us. (After giving the file .html extension)

- MDN: Mozilla Developer Network (https://developer.mozilla.org/en-US/)

- `<html>` tab is the root of an HTML file, it is followed by `<head>` and `<body>` tags

- `<head>` tag, provides general information; so basically everything that we don't see on the paper as a user, things like stylesheet files, js files; they all go in the head. In conclusion, here is where we put metadata

- `<body>` is where we put all our content

- *<!-- This is a comment -->*, simply to fastly create a comment, select the content or go on a line and hit 'Ctrl + /'

- Although `<title>` won't show up on our page, it's important because it is the text that goes up to the tab name. And pay attention to the reason it won't show up on the page is it is in the head/metadata section. So it is also the name that shows up on the google results

- MDN Element Reference: Lists all the elements of HTML (https://developer.mozilla.org/en-US/docs/Web/HTML/Element) around 100 or so but 15-20 of'em frequently used

- `<h1>` is the most, `<h6>` is the least important heading... Tags like this headings called as 'block level elements' which basically means even we write'em on a single line they are seperately shown on lines on the webpage

- 'Inline elements' are the ones which if written on the same line they are shown on the same line.

- `<p>` Paragraf tag makes the text block level

- `<b>` tag is the basic way to bolden a text; this is inline also; but there is a better tag to do it which is `<strong>` tag. Although it does exactly the same thing it emphasizes the importance of it in the source code -as far as I guess- and things like these tags called 'semantic mark' in HTML5

- `<i>` makes the text italic, it's inline; like the above one, `<em>` tag makes the same but a better way in terms of semantic marks

- `<ol>` tag creates an ordered list, which means every item takes number in front of it as 1,2,3 etc, and inside this list we have to specify every one of the elements with `<li>` tags

- `<ul>` tag creates an unordered list, which just takes bulletpoints in front of'em, again inside this list we have to specify every one of the elements with `<li>` tags

- we can make lists in lists

- Ctrl + Shift + D, automatically copies the current line and pastes on next line directly.

- First headings, outer lists, inner lists

- Inside an html file, typing "lorem" and hitting Tab right after it, is going to give a long text to us, automatically.

- `<div>` is a way of grouping things together. But actually it won't give us much benefit until we get to CSS. Because then we will go and style that group together. It is a block level element btw, which means inner divs will cause new lines

- `<span>` is also a generic container but there is a key difference is an inline one. So we will use this one inside of a div if we want to give that part a more specific feature along with the div features.

- Attributes: adding additional information to tags. Its form is like the following, a key-value pair,

```
  <tag name="value"></tag>
```

- `<br>` tag is a break line tag, which passes you to the next line

- We don't need a closing tag for `<img>`, `<br>` attributes.

- Anchor Tag is the one we all see in every Google search result actually, usage:

```
  <a href="url">Link Text</a>
```

I need to make my link explicit at this point. So I have to add that "http://" part at the beginning otherwise it tries to find that following part at my current folder!

- Table usage:

```
  <table>
   <thead>
     <tr>
      <th>Heading for 1st column</th>
      <th>Heading for 2nd column</th>
     </tr>
   </thead>
   <tbody>
     <tr>
      <td>1st row 1st column</td>
      <td>1st row 2nd column</td>
     </tr>
   </tbody>
  </table>
```

tr used for creating rows, td for columns, th for headings
`<table border="1">` can be used to determine the borders but it makes more sense to do it with CSS
`<thead>` and `<tbody>` tags are like the head and body tags of the page, it is optional but better to use
`<img width="">` can be used for reararranging the size of an image but actually we will be doing this by CSS later on. PS, it automatically scales the image by the given width

- In order to create multiple cursors in Sublime, simply hold Ctrl and click on places that we want the cursor to be on

- HTML Forms: Getting input from user:

- `<form></form>`: It's a shell/container for inputs like username, mail, password, log in button, dropdown menus, radio buttons etc. So by just forming a 'form' we don't actually create something visual, all buttons etc. must be written explicitly in this for something visual.

The following attribute pair sends data from the form to a server somewhere. Action atr contains the URL to send form data to and Method specifies the type of the HTTP request. We use 'get' request when we try to retrieve some data like searching etc. and we use 'post' request when we are sending data like wanting it to be added to a database or posted to a server. i.e. Google search is get request; Facebook signup is post request:

```
  <form action="/my-form-submitting-page" method="post"> ALL THE INPUT </form>
```

By default if we don't specify the URL part it is the current location. If we don't specify the method it's a 'get' request by default.

- `<input>` tag creates interactive controls.
The "type" atr determines the types of input.

```
<input type="text/date/color/file/checkbox/password/radio/...">
```

The "name" atr specifies the key of the current input in the querystring.

    <input name="username" type="username">

The "placeholder" atr is the default text inside a box

    <input name="username" type="username" placeholder="username">

- Ctrl + L select the current line

- `<label></label>`: We put `<input>` tags into this to give them a label and make them more comprehensive:

```
<label>
  Username:
  <input name="username" type="username" placeholder="username">
</label>
```

An alternative and a better way to use `<label>` tag with `<input>` tags is as the following, by using "for" and "id" ATRs. By doing so we prevent nests:

```
  <label for="username">Username:</label>
  <input id="username" type="text" placeholder="username">
```

In order to add some validation, we can first use to check the presence of some input, "required" label. When we change type to "email" for example, it will automatically inspect the input is approppriate for an email or not.

```
  <input id="email" name="email" type="email" placeholder="email" required>
```

- Radio Boxes:

Regarding radio boxes, actually we can select more than one in radio button but cannot unselect it so we have to arrange it manually to make exactly one choice, to do this we give the choices the same "name"!, but here we must also specify the "value" of the choice for a better info sending and a better query; Used in things like gender selection etc. (Don't forget to add Labels)

```
<input id="male" name="gender" type="radio" value="male">
<input id="female" name="gender" type="radio" value="female">
```

- `<input type="checkbox">`

A checkbox is the box which we can make a tick or not like selecting our skills, multiple or none

- A `<button></button>` at the end of a form will simply submit the form

- Select Tag:

With this tag we can create a nice dropdown selection menu. Usage:

```
<select name="color">
  <option>Red</option>
</select>
```

Here because we didn't specify a value, it will automatically set the value in the query to the string on the option which is "Red" here. But sometimes we may want to specify it explicitly, to do so:

```
<select name="mood">
  <option value="Happy">:)</option>
</select>
```

Now the value will be set to "Happy" in the query if we select ":)"

- Textarea Tag:

This tag is used for inputs that are more than one line. "rows" & "cols" ATRs can be used to specify the size of the box,

- Pattern & Required Title ATRs,

```
   <input id="password" type="password" name="password" pattern=".{5,10}" 
required title="errorMsg">
```

We can type as much or as less as we want but cannot submit due to the pattern and the title (by adding "Please match the requested format" on the above line of our title) will be shown.

But if we use the following,

```
   <input id="password" type="password" name="password" minlength="5" 
maxlength="10" required title="errorMsg">
```

Here it restricts the input between 5 to 10. We cannot even type nor less than 5 neither more than characters. And the title won't be showed up to the user.